N92-24405

# Algebraic Surface Grid Generation in Three-Dimensional Space

Saif Warsi
Sverdrup Technology, Inc.
Structures and Dynamics Dept.
Huntsville, Alabama 35806

## SUMMARY

An interactive program for algebraic generation of structured surface grids in three-dimensional space has been developed on the IRIS4D series workstations. Interactive tools are available to ease construction of edge curves and surfaces in three-dimensional space. Addition, removal, or redistribution of points at arbitrary locations on a general three-dimensional surface or curve is possible. Additionally, redistribution of surface grid points may be accomplished through use of conventional surface splines or a method called here as "surface-constrained transfinite interpolation". This method allows the user to redistribute the grid points on the edges of a surface patch; the effect of the redistribution is then propagated to the remainder of the surface through a transfinite interpolation procedure where the grid points will be constrained to lie on the surface. The program has been written to be highly functional and easy to use. A host of utilities are available to ease the grid generation process. Generality of the program allows the creation of single and multi-zonal surface grids according to the user requirements. The program communicates with the user through popup menus, windows, and the mouse. General picking mechanisms have been implemented so the user does not have to keep track of curves or surfaces through any sort of identification mechanism external to the program. Further, this program has been written in the C language to allow use of dynamic memory allocation so that the required memory during grid generation grows as needed.

## INTRODUCTION

The growing capabilities of modern computers are driving engineers to apply their algorithms to increasingly more complex geometries. As a consequence, traditional grid generation procedures using batch grid generation and plotting programs are no longer suitable tools to provide acceptable meshes in a reasonable time. For any highly complex geometry, generation of acceptable surface grids may consume more time than obtaining the flow solution itself.

To meet the demands of obtaining reasonable surface grids in a timely manner, an interactive program using the graphical capabilities of todays high speed engineering workstations has been developed. The interactive approach is reliable and timely because it allows the user to correct errors without delay. Due to the interactive nature of the program, the results of

any operations performed on curves and surfaces are immediately visualized and immediate steps may be taken to correct the final results.

The variety of different configurations which may be considered may range from simple airfoils to complex three-dimensional geometries of a whole aircraft, nozzles, inlets, etc. Therefore the grid generation techniques have been designed to be flexible, general and easy to use. To meet the above requirements this program allows arbitrary geometries to be prepared, generated, and refined interactively through a user interface consisting of popup-menus and windows; generally the user has the use of the entire graphics window. General algorithms and data structures have been implemented to ease the grid generation process. The program has been written to be an "easy to use" tool for constructing curves and surfaces without having to be a grid generation "expert".

## EDGE CURVE GENERATION

The program has extensive edge curve generation and editing capabilities. Edge curve generation in the current program may be accomplished through extraction of interactively specified grid lines from an existing surface, interactively designed straight lines, circular and elliptical arc segments or three-dimensional bézier curves. The circular and elliptical arc segments may lie in any arbitrary three-dimensional plane. End points for any curve may be either interactively specified, input or picked from any existing surface or curve. To aid endpoint selection, cursor movement to arbitrary points in three-dimensional space is easily done without the user having to do complex rotations or translations. Additional curves may be generated by scaling, translating, appending to, rotating any existing curve or splitting existing curves into multiple segments as specified by the user. The rotation of curves may be done about any user-specified axis.

The bézier curves [1] are an especially useful edge curve generator tool since a wide variety of curves may be easily designed.

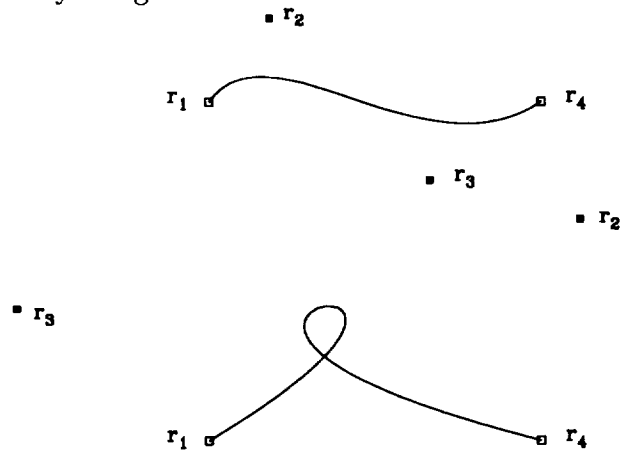


Figure 1. Control point location influence on bézier curves.

Slopes at the end points and overall curve shape are entirely under user control. The bézier curve may be written in matrix notation as

$$\mathbf{r}(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \\ \mathbf{r}_4 \end{bmatrix} \tag{1}$$

where $u$ is a parameter with $0 < u < 1$ and the $\mathbf{r}$'s are the coordinate vectors of control points defining the curve. Referring to Figure 1, $(\mathbf{r}_1, \mathbf{r}_4)$ are the endpoints of the curve and $(\mathbf{r}_2, \mathbf{r}_3)$ are the control points whose locations guide the shape of the curve. The ($\mathbf{r}_2$ and $\mathbf{r}_3$) control points are entirely under user control and may be moved to any location in three-dimensional space to obtain the desired curve. As figure 1 shows, the influence of the location of the control points allows a variety of curves to be easily generated.

To ease construction of edge curves for complex three-dimensional geometries, a bézier curve with either endpoint $\mathbf{r}_1$ or endpoint $\mathbf{r}_4$ being on a surface will have its control points located in three-dimensional space such that the initial curve will be orthogonal to the surface at that endpoint.

Curve editing facilities include redistribution and addition/deletion of points along a given user specified segment of a curve. These procedures use parametrized cubic splines to either redistribute or add/delete points along the specified segment. Redistribution of grid points uses the Vinokur [2] hyperbolic stretching function; the desired spacing at either end of the specified segment may be either input by the user or interactively adjusted to user requirements. As there are no restrictions in this method, any curve can be given the proper number of points with desired distribution characteristics. Curves which have a discontinuity can be handled easily by placement of an endpoint of the chosen segment at the discontinuity such that spline construction and evaluation do not occur across the discontinuity itself.

## SURFACE GRID GENERATION

The surface generation and editing package in the program is also quite general. Initial surfaces may be read from a file or created within the program. Surfaces may be created using the following methods:

1. Arc-Length based transfinite interpolation given the edge curves (one collapsed edge is possible).
2. Surface of revolution by rotating a given curve about any user-specified axis.
3. Automatic splining of a given set of cross sections.
4. Rotation of a surface about any user-specified axis.
5. Scaling, translating and mirroring a surface.
6. Appending surfaces to create a new surface.

To create a surface given its edge curves, the transfinite interpolation ($TFI$) method of Soni [3] is used to calculate an initial interior surface grid. The algebraic grid generated via $TFI$ may require smoothing to remove discontinuities; future plans include implementing the interactive elliptic smoothing package of Kania [4]. This grid generator solves the most general elliptic equations for surfaces (see ref. [5]) using a line successive over-relaxation (LSOR) method and maintains the surface curvature.

A surface of revolution may be easily generated by revolving a given curve about any arbitrary user-specified axis. The user simply specifies the number of points in the rotation direction and either inputs or interactively specifies the rotation angle. A surface may also be easily created by splining a set of specified cross sections. The user picks the desired cross sections and specifies the number of points through the sections. The code will automatically place a uniformally distributed surface through the sections, however, grid lines on the generated surface will not neccessarily lie exactly on the original cross sections except at the first and last specified cross sections.

The surface editing package has similar features as the curve editing package. Addition or deletion of grid points may be performed easily by interactively drawing a patch on a given surface (the patch may be as small as one cell wide) and specifying the number of points to add or remove. Figure 2 shows a generic forebody surface grid of an aircraft; figure 3 shows a case where the number of points was increased in the canopy region. Discontinuites in a surface, as for a curve, may be handled easily by selecting appropriate patches so that spline constructions and evaluations do not occur at the discontinuites.

Redistribution of grid points on a surface may be performed in two modes. The first mode allows the user to specify any portion of a constant $\xi$ or $\eta$ line on a given surface. The grid points may then be redistributed using parametric surface splines. For example, redistributing a portion of any constant $\xi$ line will cause all grid points in the specified range of $\eta$ on each $\xi$ constant line to also be redistributed; similarly a constant $\eta$ line may be selected along which to redistribute. Returning to the forebody surface grid, let the $\xi$ coordinate be along the body and let $\eta$ be the transverse coordinate. Figure 4 shows an application of this mode; notice that the grid points have been redistributed to the top of the canopy in the $\eta$ direction and near the end of the body and from the nose to just ahead of the canopy in the $\xi$ direction.

The second mode allows the user to choose a portion of a surface as a surface patch and a new system of coordinates ($\xi_1$,$\eta_1$) may then be generated which will be constrained to line on the original surface. This mode allows the user to make local grid modifications without disturbing the grid distributions elsewhere on the grid. Choosing a portion of a surface as a patch, the edges of the patch (both the $\xi$ constant and $\eta$ constant) may then be redistributed. To generate the grid in the interior of the patch based upon the new $\xi_1$ and $\eta_1$ coordinates on the edges, a standard transfinite interpolation may be applied if the surface is planar or constraint of the new coordinates to the previous $\xi$ and $\eta$ system is not required. However, if constraining the new coordinates to lie on the previous $\xi$ and $\eta$ system is required, a new method, called here "surface-constrained transfinite interpolation" (SCTI) may be applied.

Description of the SCTI method

The SCTI method incorporates a bicubic spline interpolating technique. Referring to figure 5, the method first constructs and stores parametric cubic splines $C_\xi(s_\xi)$ along each $\eta$ constant line of the patch based on the original, unmodified surface (Basis Grid). The $C$ indicates the cubic spline and $s$ is the normalized arclength which is used as the parameter, in the subscripted ($\xi$) direction. A transfinite interpolation procedure is then applied

$$TFI \Rightarrow s_{\xi_1}(\xi_1, \eta_1) \text{ and } s_{\eta_1}(\xi_1, \eta_1),$$

where $s_{(\ )}$ indicates unit arclength distributions based on the new $\xi_1$ and $\eta_1$ edge coordinates. Using the $s_{\xi_1}(\xi_1, \eta_1)$ distribution the new coordinates are evaluated from the splines $C_\xi(s_\xi)$. A series of one-dimensional parametric splines $C_1$ are then constructed based on these new coordinates and the $s_{\eta_1}(\xi_1, \eta_1)$ distribution is used to evaluate the $C_1$ splines for the final surface constrained coordinates (SCTI Grid). The SCTI method implemented in the program can also handle singular patch edges.

The SCTI method discussed here is equivalent to the differential method discussed in [6]. In ref. [6] two partial differential equations (PDE's) have to be solved in which the dependent variables are the new coordinates ($\xi_1$,$\eta_1$) and the independent variables ($\xi$,$\eta$) are the old coordinates. It seems that the present algebraic approach is much faster and computationally simpler than the PDE approach.

Figure 6 shows a rather extreme example of SCTI applied to the forebody surface grid. A patch was selected and the grid point distributions on its edges were modified such that extreme coordinate concentration would occur near the corners; the SCTI method was then used to generate the new coordinates. Notice that the new coordinates are constrained to the original surface and that the influence of the edge redistributions has been propagated to the remainder of the surface by selecting appropriate surface patches and then applying SCTI.

## DATA STRUCTURES

Curve or surface data, read in from a file or constructed within the program, is stored in structures consisting of one-dimensional arrays. All storage required is dynamically allocated, reallocated or freed as needed; this feature allows users to have various size curves and surfaces in memory simultaneously. The spline, transfinite interpolation, and various other routines require temporary storage to complete their operations; this temporary storage is also allocated and then freed upon completion of the required operation.

Dynamic memory management alleviates the user from having to stop and recompile the program simply because the amount of memory allocated at compile time may not have been sufficient to complete the required grid(s). The program will warn users and suggest steps to continue the grid generation process if the user should exceed the memory limitations of the machine.

## APPLICATIONS

Figure 7 shows a set of curves generated for a generic re-entry vehicle as an example of curve generation and editing facilities. Since actual surface definition data was not available, each of the curves shown was generated with the curve segment generator in the program. The majority of the curves were generated using the bézier curve generator however, the complex curves at the trailing edge of the wing were generated by appending multiple bézier curves, elliptical, circular and straight line segments.

Figure 8 shows the initial surface grid generated for the generic re-entry vehicle using the previously designed curves shown in figure 7 and the surface generation facilities of splining cross-sectional data and transfinite interpolation with specified edge curves. Figure 9 shows the final surface grid for the generic re-entry vehicle after using the surface editing facilities. Notice that grid distributions are now much smoother and point resolution in areas of interest is better, while the original surface geometry is maintained.

Figure 10 shows a sample far field boundary and blocking arrangement for the re-entry vehicle after performing a domain decomposition. Figure 11 shows the mesh on selected block faces around the re-entry vehicle (coarsened grids are shown for clarity).

Figure 12 shows a global view of the surface grids generated for a wing/pylon/load configuration. Figure 13 shows some detail of the surface grids in the wing/pylon intersection region. Once again, all of the initial and final curves and surfaces were generated entirely within the program.

The interactive approach to surface and edge generation, as implemented in this program, is very efficient. For instance, generation of the initial curves for the re-entry vechicle was the most time consuming portion of the grid generation process since vehicle surface definition data was not available. The process of generating the initial curves was iterative so that a "proper" vehicle could be constructed; this process consumed approximately five working days on a part-time basis. The creation of the initial surfaces given the curves required approximately ten minutes. Editing the initial surfaces, domain decomposition and generation of the outer boundary blocking structure and surfaces took approximately two hours, since this procedure was also iterative.

## CONCLUSIONS

An interactive algebraic grid generator has been developed which is capable of generating the surfaces for general three-dimensional geometries. The grid generator allows users to easily decompose complex domains and create any three-dimensional edge curve. Various utilities have been developed to ease coordinate generation on both curves and surfaces. The functionality, versatility, and efficiency of the interactive approach to surface generation has been demonstrated.

## FUTURE WORK

Earlier, an interactive code for single block, planar grids, called GEN2D [*] was released. A very useful feature in GEN2D is the ability of the user to reshape any portion of any grid line via bézier curves. The effect of the reshaped grid line can then be propagated by $TFI$ to the remainder of the grid. Since obtaining a desired grid using elliptic smoothing is quite difficult in highly complex or very coarse regions, this method is a quick and cheap way to obtain desired grids even in highly complex and/or coarse regions. A similar feature is now being planned for the current program. Therefore, a method to reshape grid lines on general three-dimensional surfaces using "surface constrained bézier curves" is being formulated. With the use of the SCTI method (or a variant thereof), the effect of the reshaped grid line(s) may be introduced into the remainder of the domain.

As mentioned previously, the interactive elliptic smoothing package of ref. [4] is also to be implemented into the program after re-coding the package from FORTRAN to C.

## REFERENCES


[1] I.D. Faux and M.J. Pratt. *Computational Geometry for Design and Manufacture*, Ellis Horwood, 1979.

[2] Vinokur, M., "On One-Dimensional Stretching Functions for Finite-Difference Calculations", NASA-CR-3313, October 1980.

[3] Soni, B.K., "Two- and Three-Dimensional Grid Generation for Internal Flow Application of Computational Fluid Dynamics", AIAA Paper No. 85-1526, 1985.

[4] Kania, L., "Elliptic Surface Grid Generation in Three-Dimensional Space", In *NASA Workshop on Software Systems for Surface Modeling and Grid Generation*,NASA CP-3143, April 1992.

[5] Warsi, Z.U.A., "Numerical Grid Generation in Arbitrary Surfaces through a Second-Order Differential-Geometric Model", *Journal of Computational Physics*, Vol. 64 No. 1, May 1986.

[6] Warsi, Z.U.A., "Theoretical Foundation of the Equations for the Generation of Surface Coordinates", *AIAA Journal*, Vol. 28 No. 6, June 1990.

[*] Warsi, S.A., "User's Guide to GEN2D - An Interactive Program for Generating Simply Connected Grids", Sverdrup Technology, Huntsville, AL. 1990, (unpublished).

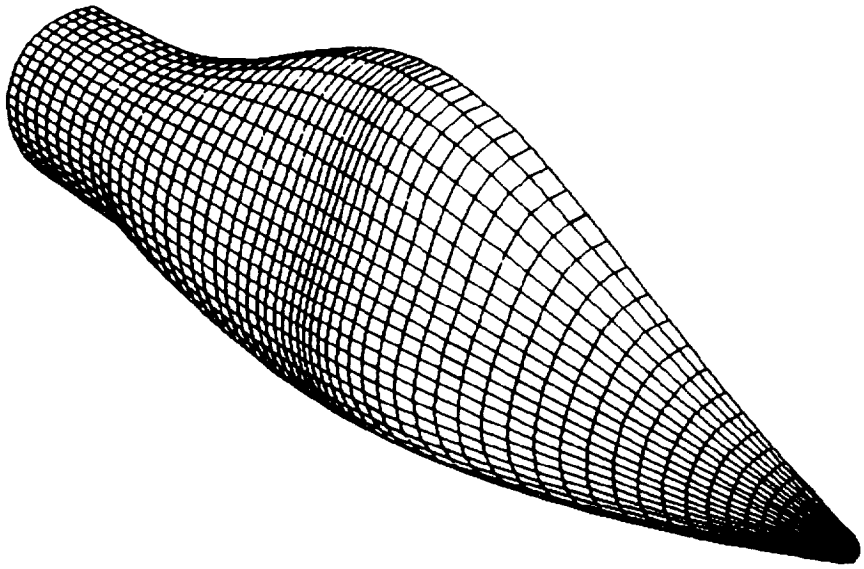

Figure 2. Generic forebody surface grid.

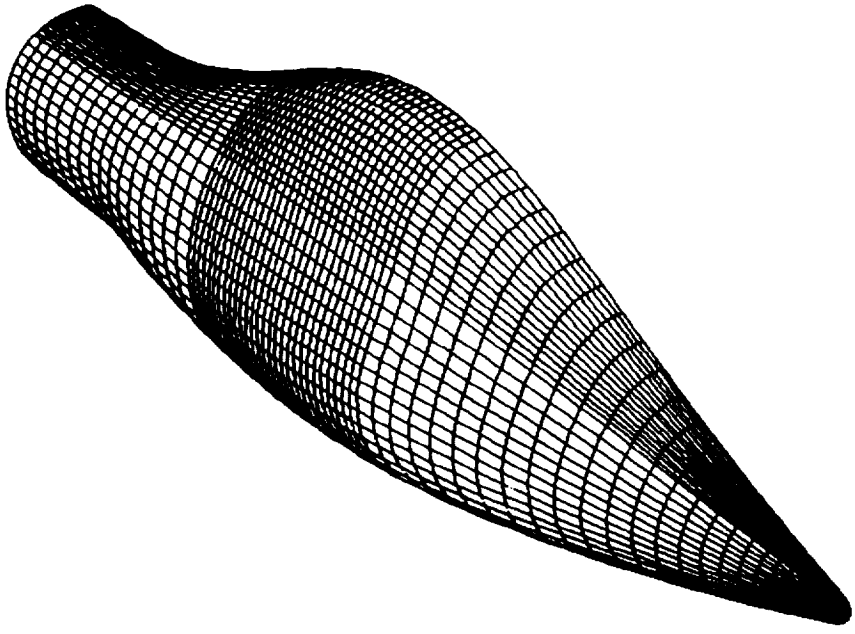

Figure 3. Enrichment of grid points in canopy region of forebody surface.

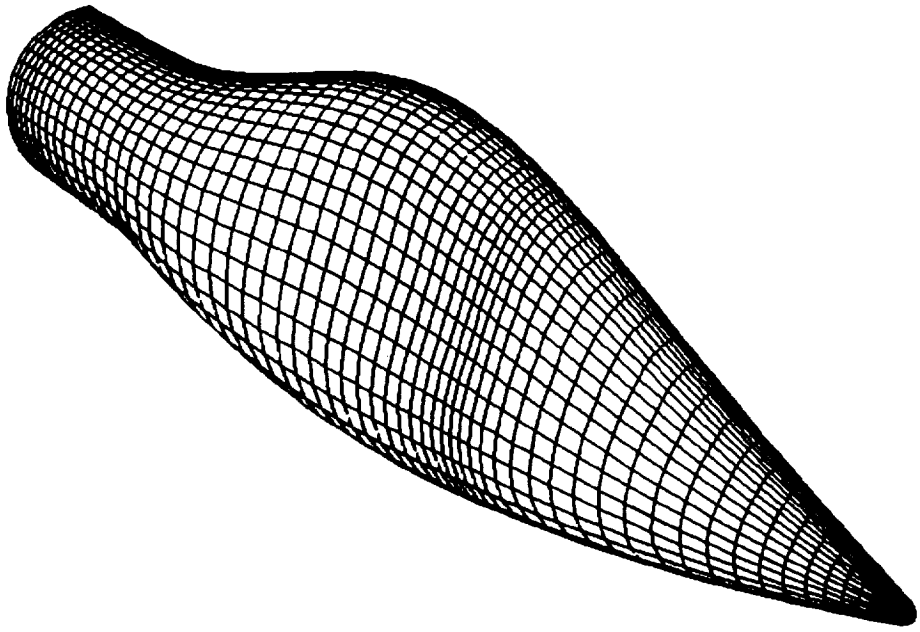

Figure 4. Redistribution of grid points on forebody surface using conventional splines.

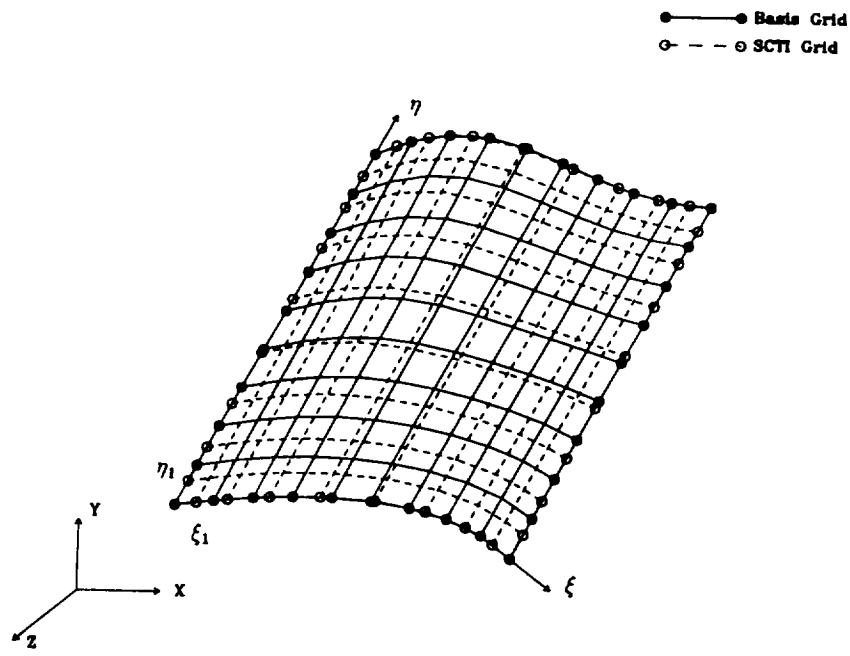


Figure 5. Concept of the SCTI method

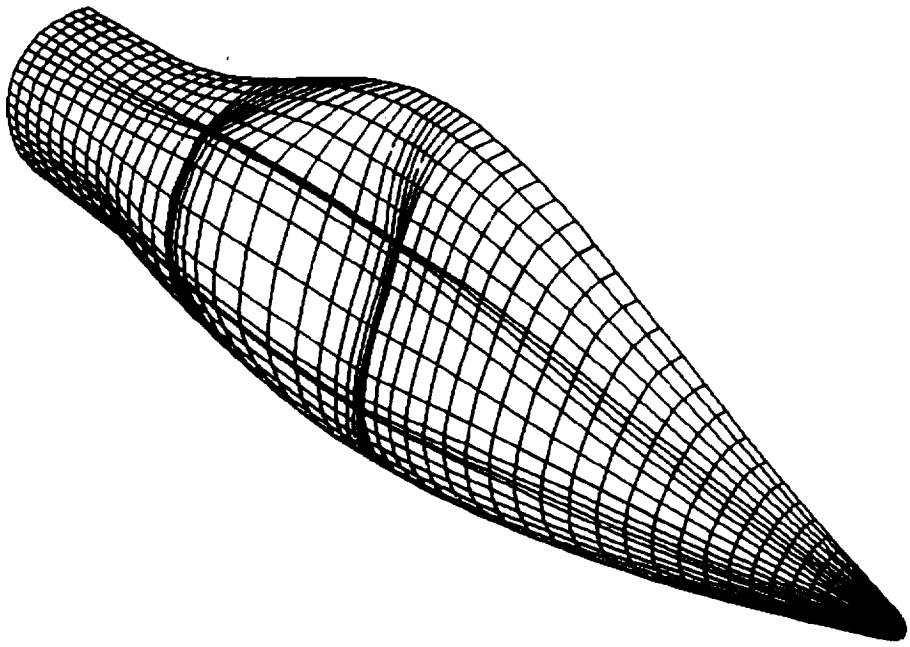

Figure 6. Application of SCTI method to forebody surface grid

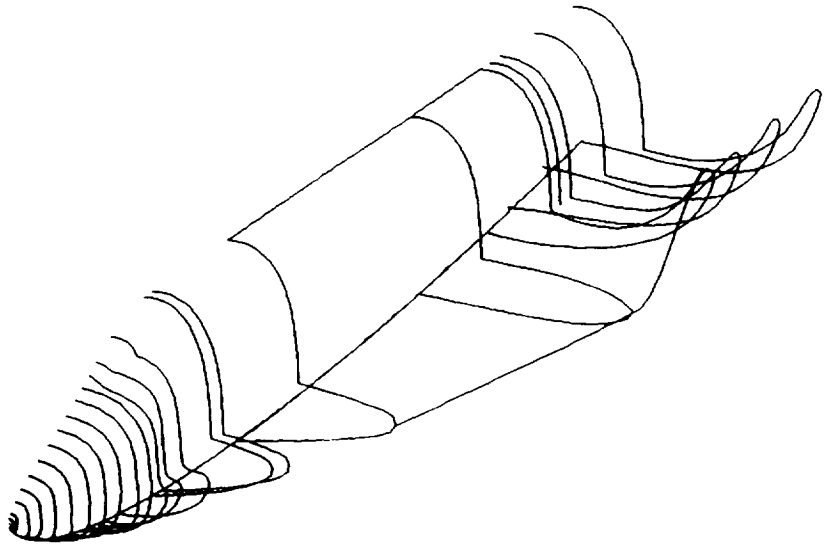

Figure 7. Surface definition curves for generic re-entry vehicle.

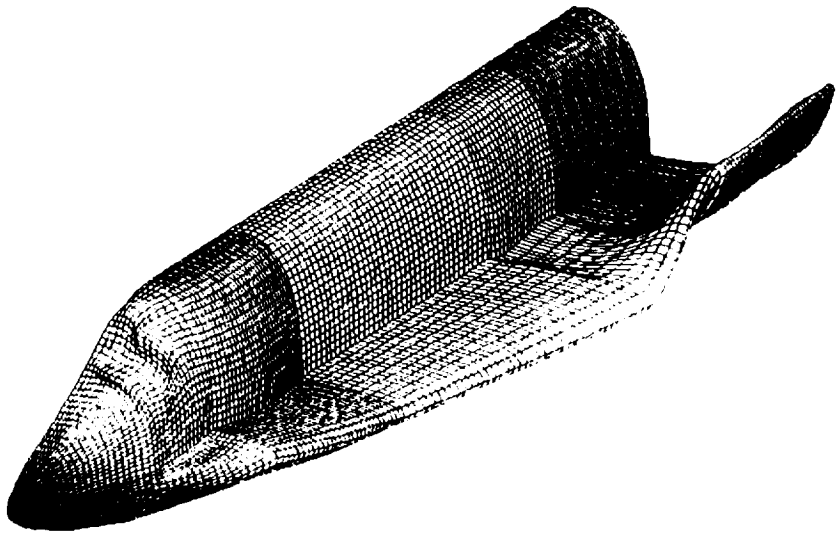

Figure 8. Initial surface grid for generic re-entry vehicle.

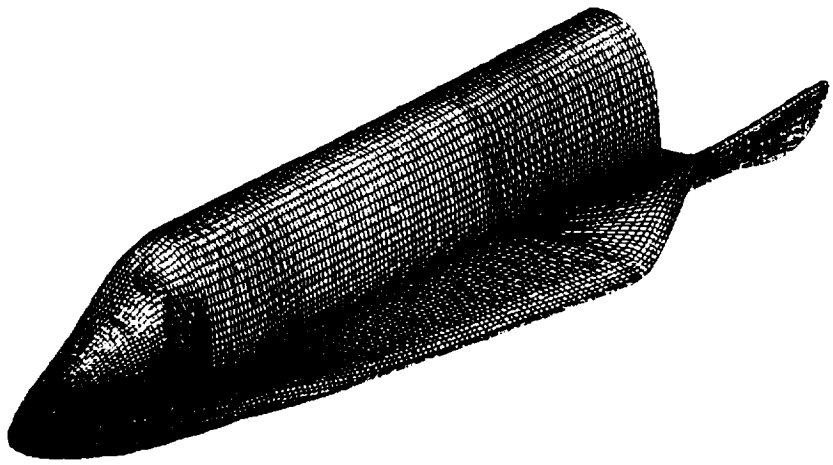

Figure 9. Final surface grid for generic re-entry vehicle.

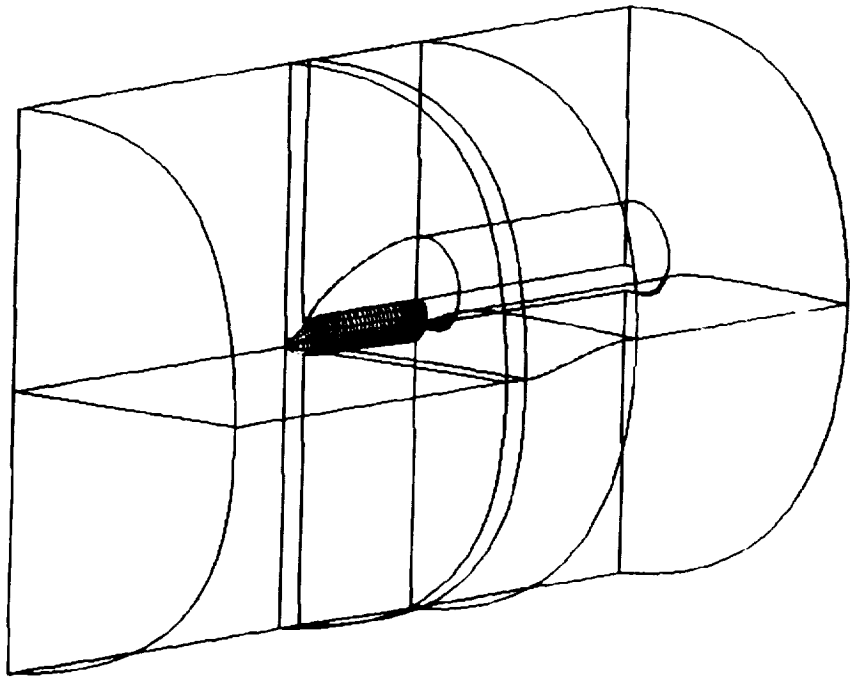

Figure 10. Example farfield and blocking arrangement for re-entry vehicle.

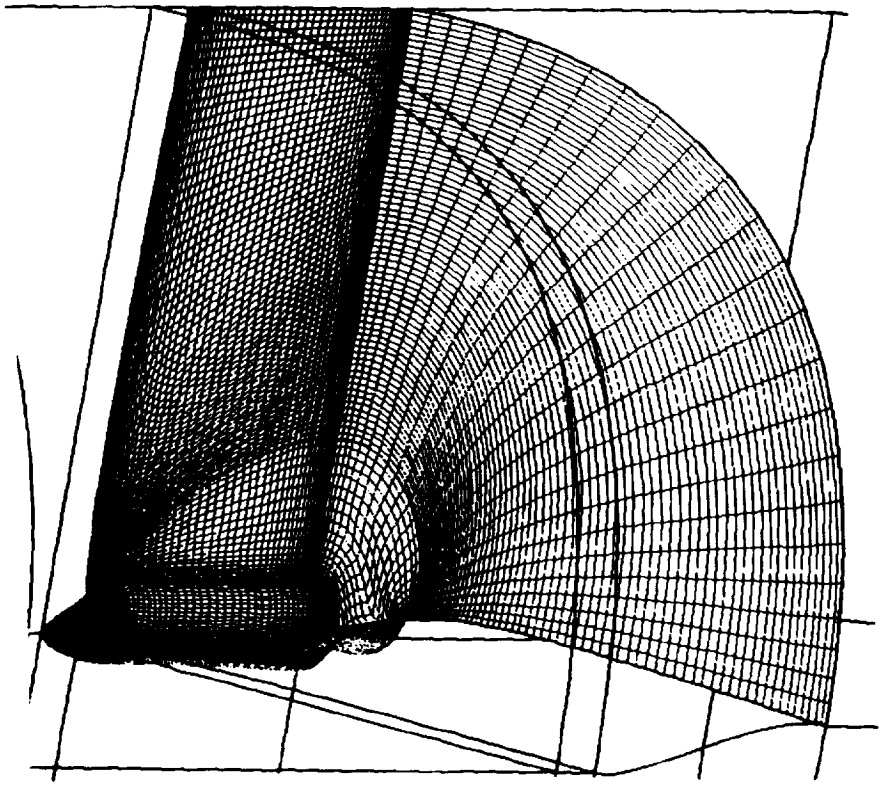

Figure 11. Grids on block faces around re-entry vehicle.

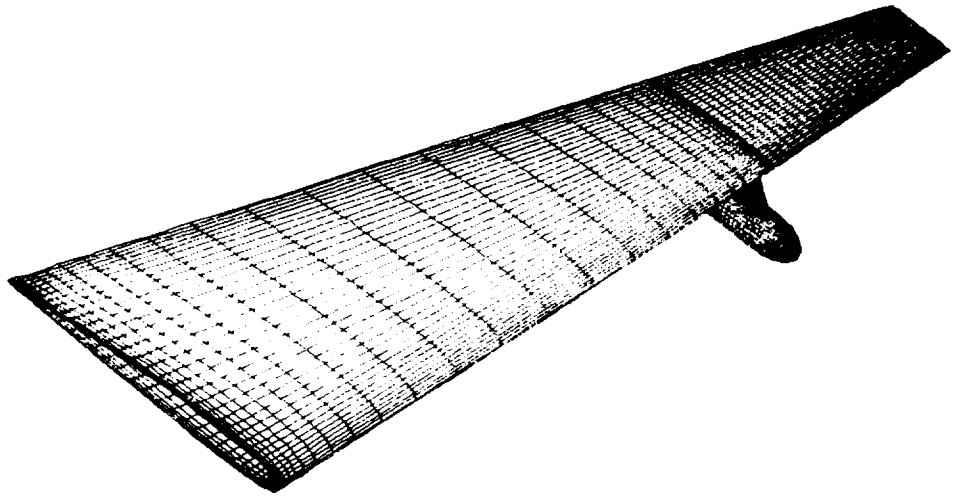

Figure 12. Surface grids for wing/pylon/load configuration.

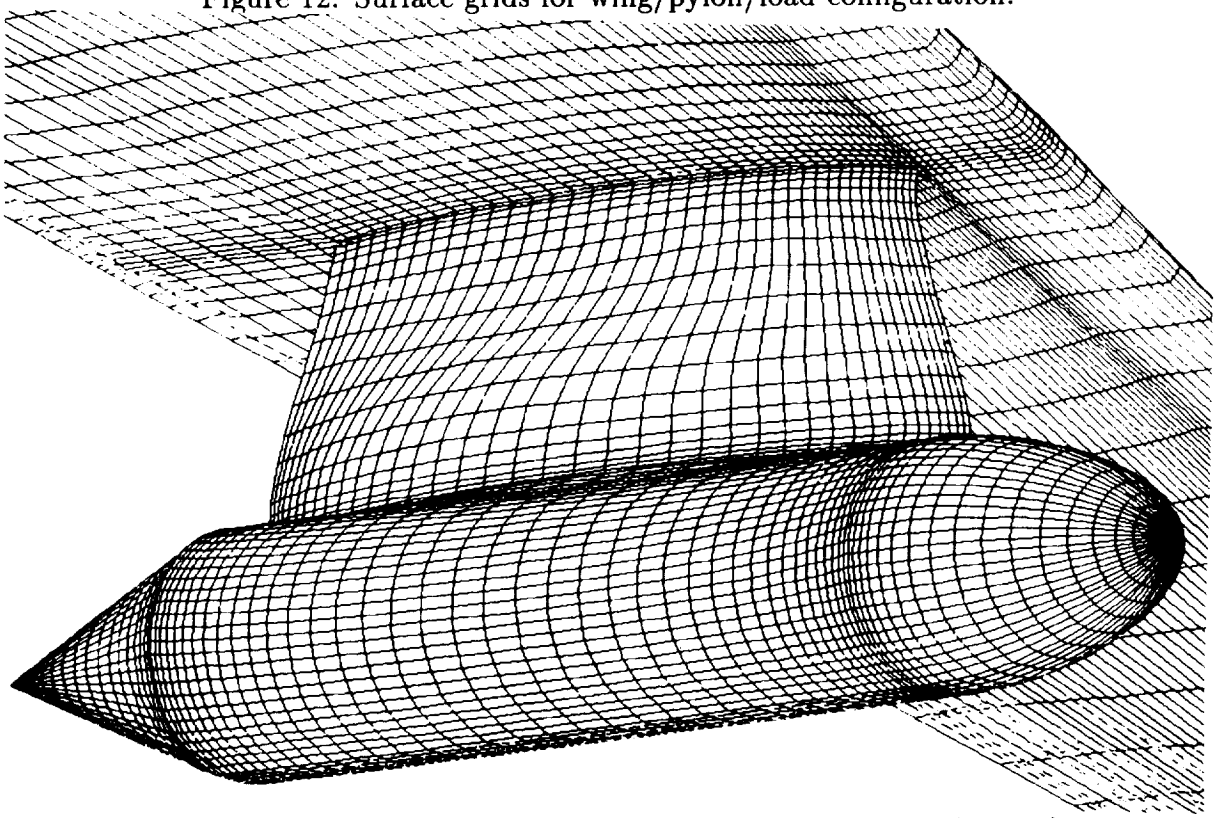

Figure 13. Detail of surface grids in the wing/pylon intersection region.